In [1]:

```
# Download Feature Enginering Data
#!wget --header="Host: doc-0s-0c-docs.googleusercontent.com" --header="User-Agen
```

In [2]:

```
# Load All The Libraries

import pandas as pd
import numpy as np
import random
from  datetime import datetime, timedelta
import joblib
```

In [3]:

```
# Read The Data

data = pd.read_pickle("Data_m5.pkl")
```

In [4]:

```
data.shape
```

Out[4]:

```
(27023821, 47)
```

1. Choose Any Item ID Randomly

In [5]:

```
m1 = random.randrange(data.shape[0])
unique_id = data['id'][m1]
unique_id
```

Out[5]:

```
'FOODS_2_137_TX_3_evaluation'
```

In [6]:
```
data_point = data[data['id']==unique_id]
data_point
```

Out[6]:

	id	item_id	dept_id	cat_id	store_id	state_id	d	sales
16995	FOODS_2_137_TX_3_evaluation	1887	5	2	6	1	d_1050	0.0
144994	FOODS_2_137_TX_3_evaluation	1887	5	2	6	1	d_1051	0.0
144995	FOODS_2_137_TX_3_evaluation	1887	5	2	6	1	d_1052	0.0
144996	FOODS_2_137_TX_3_evaluation	1887	5	2	6	1	d_1053	0.0
144997	FOODS_2_137_TX_3_evaluation	1887	5	2	6	1	d_1054	0.0
...	...	...	...	...	...	...	...	..
26891214	FOODS_2_137_TX_3_evaluation	1887	5	2	6	1	d_1965	NaN
26891215	FOODS_2_137_TX_3_evaluation	1887	5	2	6	1	d_1966	NaN
26891216	FOODS_2_137_TX_3_evaluation	1887	5	2	6	1	d_1967	NaN
27003355	FOODS_2_137_TX_3_evaluation	1887	5	2	6	1	d_1968	NaN
27003356	FOODS_2_137_TX_3_evaluation	1887	5	2	6	1	d_1969	NaN

920 rows × 47 columns

2. Downcast The Data

In [7]:
```
#https://www.kaggle.com/anshuls235/time-series-forecasting-eda-fe-modelling/notel

def downcast(df):

    cols = df.dtypes.index.tolist()
    types = df.dtypes.values.tolist()

    for i,t in enumerate(types):

        if 'int' in str(t):
            if df[cols[i]].min() > np.iinfo(np.int8).min and df[cols[i]].max() <
                df[cols[i]] = df[cols[i]].astype(np.int8)
            elif df[cols[i]].min() > np.iinfo(np.int16).min and df[cols[i]].max(
                df[cols[i]] = df[cols[i]].astype(np.int16)
            elif df[cols[i]].min() > np.iinfo(np.int32).min and df[cols[i]].max(
                df[cols[i]] = df[cols[i]].astype(np.int32)
            else:
                df[cols[i]] = df[cols[i]].astype(np.int64)

        elif 'float' in str(t):
            if df[cols[i]].min() > np.finfo(np.float16).min and df[cols[i]].max(
                df[cols[i]] = df[cols[i]].astype(np.float16)
            elif df[cols[i]].min() > np.finfo(np.float32).min and df[cols[i]].ma
                df[cols[i]] = df[cols[i]].astype(np.float32)
            else:
                df[cols[i]] = df[cols[i]].astype(np.float64)

        elif t == np.object:
            if cols[i] == 'date':
                df[cols[i]] = pd.to_datetime(df[cols[i]], format='%Y-%m-%d')
            else:
                df[cols[i]] = df[cols[i]].astype('category')

    return df
```

In [8]:
```
data = downcast(data)
```

3. Preprocessing The Data

In [9]:
```
def preprocessing(data):
    """
     to preprocees the data
    """

    # A.  Get integer value in d column    ex d_1 , d_2   ---->>>   1  ,1
    data['d'] = data['d'].apply(lambda a: a.split('_')[1]).astype(np.int16)

    # B. Filter The Data
    dt = data[data['d']>=1871]
    return dt
```

4. Feature Engineering The Data

In [10]:
```
def feature_engineering(dt):
    lags = [7, 28]
    lag_cols = [f"lag_{lag}" for lag in lags ]
    for lag, lag_col in zip(lags, lag_cols):
        dt[lag_col] = dt[["id","sales"]].groupby("id")["sales"].shift(lag)

    wins = [7, 28]
    for win in wins :
        for lag,lag_col in zip(lags, lag_cols):
            dt[f"rmean_{lag}_{win}"] = dt[["id", lag_col]].groupby("id")[lag_col

    return dt
```

In [11]:
```
h = 28
max_lags = 70
fday = datetime(2016,5, 23)
fday
```

Out[11]:
```
datetime.datetime(2016, 5, 23, 0, 0)
```

In [12]:
```
def final_Pred1(data_point):

  dt = preprocessing(data_point)


  useless_cols = ["id", "date", "sales","d", "wm_yr_wk", "weekday", "sell_price",
  train_cols = dt.columns[~dt.columns.isin(useless_cols)]

  # Load The Model
  m_lgb = joblib.load("lgb_Model_0.pkl")
  for tdelta in range(0, 28):
      day = fday + timedelta(days=tdelta)
      print(day)
      tst = dt[(dt.date >= day - timedelta(days=max_lags)) & (dt.date <= day)]
      tst = feature_engineering(tst)
      tst = tst.loc[tst.date == day , train_cols]
      dt.loc[dt.date == day, "sales"] = m_lgb.predict(tst)
      print("Prediction Compltete ",tdelta)
      del(tst)

  x = pd.pivot_table(dt, index='id', values='sales', columns='d').iloc[:,-28:].r

  return x
```

In [13]:
```
dt1 = final_Pred1(data_point)
dt1
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:7: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  import sys

2016-05-23 00:00:00

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: SettingWithCopy
Warning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  """
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:10: SettingWithCop
yWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  # Remove the CWD from sys.path while we load stuff.
/usr/local/lib/python3.7/dist-packages/pandas/core/indexing.py:1743: SettingWit
hCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pyd
ata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
opy)
  isetter(ilocs[0], value)

Prediction Compltete  0
2016-05-24 00:00:00
Prediction Compltete  1
2016-05-25 00:00:00
Prediction Compltete  2
2016-05-26 00:00:00
Prediction Compltete  3
2016-05-27 00:00:00
Prediction Compltete  4
2016-05-28 00:00:00
```

```
Prediction Compltete  5
2016-05-29 00:00:00
Prediction Compltete  6
2016-05-30 00:00:00
Prediction Compltete  7
2016-05-31 00:00:00
Prediction Compltete  8
2016-06-01 00:00:00
Prediction Compltete  9
2016-06-02 00:00:00
Prediction Compltete  10
2016-06-03 00:00:00
Prediction Compltete  11
2016-06-04 00:00:00
Prediction Compltete  12
2016-06-05 00:00:00
Prediction Compltete  13
2016-06-06 00:00:00
Prediction Compltete  14
2016-06-07 00:00:00
Prediction Compltete  15
2016-06-08 00:00:00
Prediction Compltete  16
2016-06-09 00:00:00
Prediction Compltete  17
2016-06-10 00:00:00
Prediction Compltete  18
2016-06-11 00:00:00
Prediction Compltete  19
2016-06-12 00:00:00
Prediction Compltete  20
2016-06-13 00:00:00
Prediction Compltete  21
2016-06-14 00:00:00
Prediction Compltete  22
2016-06-15 00:00:00
Prediction Compltete  23
2016-06-16 00:00:00
Prediction Compltete  24
2016-06-17 00:00:00
Prediction Compltete  25
2016-06-18 00:00:00
Prediction Compltete  26
2016-06-19 00:00:00
Prediction Compltete  27
```

Out[13]:

d	id	1942	1943	1944	1945	1946	1947	1
0	FOODS_2_137_TX_3_evaluation	1.094727	0.977051	0.92334	0.868652	0.868164	1.066406	1.251